Amendment to the Specification:

Please replace the paragraph beginning on page 4, line 27 with the following amended paragraph:

Generally, the SCM 12 supports managing a single SCM cluster 17~~18~~ from a single CMS 14. All tasks performed on the SCM cluster 17~~18~~ are initiated on the CMS 14 either directly or remotely, for example, by reaching the CMS 14 via a web connection 20. Therefore, the workstation 22 at which a user sits only needs a web connection 20 over a network 24 to the CMS 14 (or a managed node 16) in order to perform tasks on the SCM cluster 17~~18~~. In addition to the SCM 12 software and the HP-UX server described above, the CMS 14 preferably also comprises a data repository 26 for the SCM cluster 17~~18~~, a web server 28 that allows web access to the SCM 12 and a depot 30 comprising products used in the configuring of nodes, and a I/UX server 32.

Please replace the paragraph beginning on page 5, line 10 with the following amended paragraph:

The CMS 14 itself is preferably also a managed node 16. This is so that multi-system aware ("MSA") tools can be invoked on the CMS 14. All other nodes 16 have to be explicitly added to a SCM cluster 17~~18~~. Generally, user access to SCM 12 files is delineated to root users, who have permission to read, write, and execute files and non-root users, who have limited access to files (*e.g.*, only execute).

Please replace the paragraph beginning on page 6, line 33 with the following amended paragraph:

Figure 3 illustrates a method for generating and persisting security keys 70 according to the process shown in the sequence diagram 50 of Figure 2. As shown, the method 70 comprises entering a make key pair command 72, generating a security key pair 74~~76~~ and storing the security key pair 76~~78~~. The method 70 and the method steps 72-76~~78~~ may be executed as described above with respect to the sequence diagram 50 or differently depending on the programming environment used. Entering a make key command 72 may comprise a root user entering a CLI command to make keys. The generating a security key pair 74~~76~~ may comprise executing a method that generates a node 16 or CMS 14 public key and private key. Likewise, storing the security key pair 76~~78~~ may comprise executing a

method that serializes the public key and private key and saves the serialized key as a key file.

Please replace the paragraph beginning on page 8, line 20 with the following amended paragraph:

Figures 4a-4b illustrate a method 80 for recovering the security keys while maintaining security according to the process shown in the sequence diagram 50. As shown, the method 80 comprises entering a CLI or starting a daemon 82, reading certain security keys into a cache with root as effective UID 8688, retrieving a private key from the cache using a real UID 90, determining if the private key is successfully retrieved 92, and, if the private key is retrieved from the cache, creating a digital signature 94, sending the digital signature and a message 96, determining if the message is authorized 98, and if the digital signature is authenticated, executing an instruction 100.

Please replace the paragraph beginning on page 8, line 29 with the following amended paragraph:

Referring to Figure 4a, entering a CLI or starting a daemon 82 may comprise a non-root user entering a command at a node 16 or a daemon starting on the node 16. An authentication class (*e.g.*, Authentication) may be instantiated to enable the authentication process. Reading certain security keys into cache with root as effective UID 8688 may comprise the authentication class executing a read method that recovers certain security keys (*i.e.*, the necessary security keys for a node 16 or the CMS 14, described above) into a cache. The method 80 may also comprise setting the certain keys (not shown). Setting the certain keys 86 preferably comprises calling a method in the authentication class (*e.g.*, a setKeys method) that comprises the read method and triggers performance of the read method when called. Again, performing these steps with root as the effective UID enables the recovery of the security keys; if the root was not the effective UID, the security keys could not be recovered.